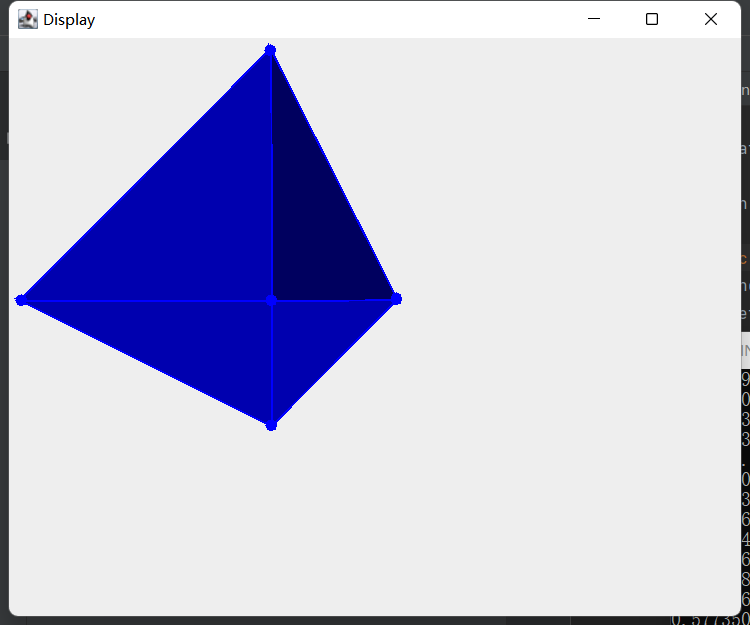
Part 1 How to use

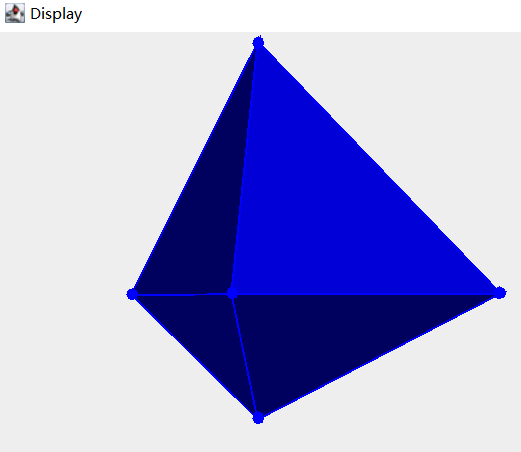
1. Edit “Software Assessment\out\object.txt”
2. Run “java -jar Software\_Assessment.jar” at the path “Software Assessment\out\”

Part 2 Result Display

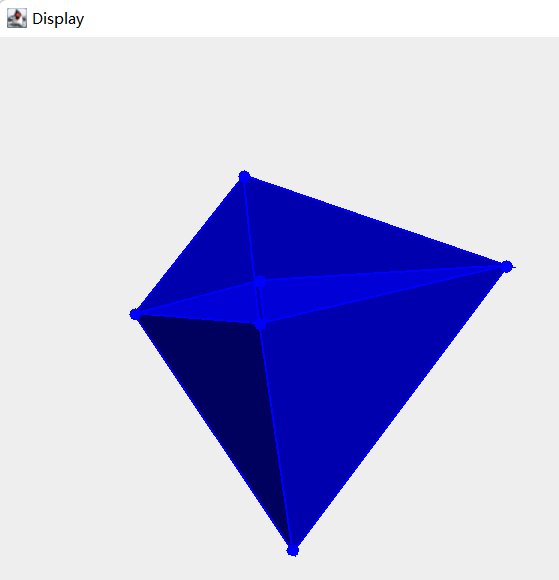
1. Initial figure



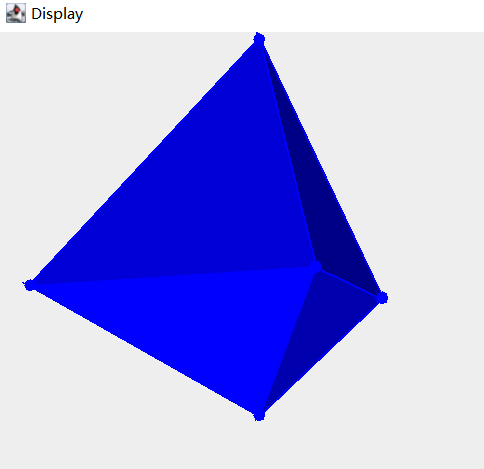
1. Rotation by Y-axis



1. Rotation by X-axis



1. Rotation by X-axis and Y-axis



Part 3 Project Structure

----root

----out

----object.txt

----Software\_Assessment.jar

----src

----Main.java

----utils

----CreateWindow.java

----Draw.java

----Face.java

----MyMouseListener.java

----ProjectedVertice.java

----TransformParameters.java

----Vertice.java

Part 4 Functionalities of each file

1. Main.java:

Run the project, create window for displaying 3D graph, and set the visibility of the window

1. CreateWindow.java:
2. Define the class of window
3. Define the constructor of CreateWindow class which sets the size, default close operation, and title of the window, calls init() of TransformParameters class to initialize the parameters, create object of Draw class to draw the graph, binds the Mouse Listener to the graph, call getInfo() of Draw class to read the input file, and adds the object of Draw class to the window.
4. Draw.java(Core)
5. Constructor-Draw(): Initialize the list used for drawing, add the color from 00005F to 0000FF in a gradient to colorList for reusing
6. getInfo(): Read “input.txt” to get the number of vertices, faces, the coordinates of vertices, the index of vertices for each face
7. paint(): In this project, 2d graphics is used to display 3d graphics, so perspective projection is chosen for the projecting. The projection is divided into twp steps: projecting the vertices(items in verticesList) from world coordinate system to view point coordinate system(items in viewPointList), and projecting them to screen coordinate system(items in projectedVerticeList). In this function, there are four parts. The first part works for the projection, the second part works for calculate the angle between the normal vector and the z-axis to decide the color of the face, the third part use Graphics.drawPolygon and Graphics.fillPolygon to draw the 3d graph. The last part draw filled circle on each vertices.
8. Face.java

Define the Face class which saves the index of three vertices which defines the face.

1. MyMouseListener.java

Listen to the event happened on the mouse, decide the direction of rotation by collecting the coordinates of mouse pointer when the mouse is pressed and released, and then the direction is calculated. Then, the parameters of transform will be updated based on direction of mouse movement. At last, it will call the repaint() to repaint the graph based on the new parameters.

1. ProjectedVertice.java

Define the class saves the information of 2d projected vertices(x and y).

1. TransformParameters.java(Method class)

Define the class saves the information of parameters used for perspective projection.

1. Vertice.java

Define the class saves the information of 3d projected vertices(x, y, z).

Part 5 Issues

Due to the complex calculation and transformation of coordinate system, there will be sometimes some issues with the direction of rotation, and the angle between Z axis and normal vector of face may go wrong sometimes, which will cause problem if viewing the 3d graph at some angles.